Faspex File Mover Post Processing Tool

This document present a specific Faspex post-processing tool that extends Faspex behaviour for simple integration into external workflow managers based on hot folders. By configuring a dropbox with special metadata (with specially formated values), files are moved to specific folders, also configured in the metadata field.

1 Prerequisites

- Faspex (4.0+) and Enterprise Server (3.7+) installed on same system
- Faspex and Enterprise Server are configured for normal package processing

2 Operating System

The configuration here is proposed for the Windows environment. The same script can also be used on Linux.

3 Ruby

The tool is developped in Ruby language. This is the language of Faspex, so the interpreter is available on all supported OSes.

A batch script (`.bat`) is used to provide the path to the ruby interpreter. If another ruby needs to be used, one can provide the path to the alternate ruby. For instance, on windows one can install ruby from: https://rubyinstaller.org/ And then set the path to ruby.exe in the batch script. The batch script provides the ruby script as argument to the ruby interpreter by changing the `.bat` extension into `.rb`.

4 Tool Installation

Choose a local folder on the Faspex system, for instance: `C:\pp`, this folder must be accessible by the user running Faspex processes (typically `svcAspera` on Windows).

Then, place the following files in this folder:

- `faspex_postprocessing.bat`
- `faspex_postprocessing.rb`
- `faspex_postprocessing.yaml`

If there is a file named `faspex_postprocessing.ba_`, then rename it to `faspex_postprocessing.bat` (gmail does not allow sending ".bat" files in email)

5 Tool Configuration

By default the tool will load its configuration file from the same folder as the script. Alternatively, another copnfiguration file can be specified as a first argument to the script.

Edit the file: `faspex_postprocessing.yaml`, it shall look like:

```
---
workflow_folder: E:/workflow
docroot: E:/
```

Note that paths shall use forward slash `/`, even on Windows. including in UNC paths like `//server/share`.

The configuration file supports the following values:

- `workflow_folder`: the main folder to which files will be moved. Subfolders specified in operations are relative to this folder.
- `docroot`: the docroot of the transfer user (typically "faspex"). The docroot can be display by executing: `asuserdata -u faspex` and then look for the line `absolute`.
- `logger`: supports value `stdout` or a file path. If no value is specified, then a log file is created in the same folder as the tool.
- `loglevel`: integer value, defaults to 1. Values are 0(DEBUG), 1(INFO), 2(WARN), 3(ERROR), 4(FATAL)

The workflow_folder and docroot shall be placed on the same drive, so that file movement are fast, else a cross drive copy would happen.

6 Faspex Configuration for the tool

In the Faspex web UI, as administrator:

- Navigate to section: Server->Postprocessing
- Click on "New"
- Give a name to the postprocessing tool: "File Mover"
- Provide the path to the batch script in section "Path to script on server" : C:/pp/faspex_postprocessing.bat
- Select "Active"
- Other parameters are left blank unless special filtering is required.
- Save

The Tool will be activated for any package exchanged on the platform. It is also possible to use the filters during script activation.

7 Tool actions

The tool is activated by creating a custom metadata option list field with values matching a special pattern (e.g. "(M:…)", at the end).

No action is executed with "Normal" packages or packages not having metadata matching an action.

If a package contains a metadata and its value ends with a part in parenthesis, this part is called "action". An example of metadata value that matches: "Blah Blah (myaction)". Here the action is "myaction". If a metadata field has no action, then nothing is done for this package.

The tool supports the following action format: `M:path/fo/subfolder`. "M" means "action move", ":" is a separator, and `path/fo/subfolder` is a sub folder path in the `workflow_folder` to which all files from the package will be moved. A file named `_FILES_HAVE_BEEN_MOVED_.txt` is created in the package (now empty) and contains information on moved files.

If Several metadata fields match the "move" action, then the destination folder is build by joining all subfolders. (if meta1 gives a/b and meta2 gives c/d, then destination is a/b/c/d). subfolder is created in the same order as metadata order in the form.

If a file from a package already exists at destination, an incremented extension is inserted between the filename and extension. For instance, if the file "f.mxf" is transfered 3 times, the destination folder will contain (in order of creation):

- f.mxf
- f.1.mxf
- f.2.mxf

Note that if a package contains many files and has an action, all files are moved, but if some of the files already existed, not all files may end up with the same "version" extension. (e.g. if a first package has "f.mxf", and a second has "f.mxf and g.mxf", the destination folder will have f.mxf f.1.mxf and g.mxf). The version number is the version of the file, not version of package.

The metadata field shall be of type "Option List". From Faspex documentation:

- List items are separated by "," (comma and space)
- An option list may be "required" or "optional".
- "optional" field has no default value selected
- "required" field has first value pre-selected
- In order to avoid a default value, and force the submitter to select a value, the list shall begin with "," (comma and space, i.e. an empty first item).

In addition, for the post-processing script to work:

- The format of each item is a text, like "Low resolution" optionally followed by an action following the format: "(M:path/to/subfolder)". Example of value for the Option List: `, Low Resolution, High Resolution (M:provider_a\ingest\highres)`

This example means:

- if submitter selects "Low Resolution", then package files remain in the package
- if submitter selects "High Resolution", then package files are moved to the subfolder `provider_a\ingest\highres` in the main folder specified by `workflow_folder`

Typically one would create a metadata profile corresponding to a type of submitter, and assign this profile to a single dropbox, and then invite submitters to this dropbox.

8 Example of Faspex Dropbox Configuration

An example requirement is the following:

- Receive various types of contents from "ProviderA"
- Low resolution content stays on Faspex as package (no movement)
- High resolution content shall arrive in a specified folder on the system: E:_a

The procedure to provision Faspex is the following:

1. Create a metadata profile specific for this dropbox.

- Navidate to: Server->Metadata, then click "Add new profile"
- Provide a name: "ProviderA_ingest". Typically the same as the dropbox that will be created in a later stage, for instance .
- Add an "Option List" metadata field (any name), and set the value to: `, Low Resolution, High`

  `Resolution (M:provider_a\ingest\highres)`
- One can remove the “Note” field, by un-selecting “Enabled”, if notes are not necessary.
- Other metadata fields may be created if necessary, but will not be processed by the tool.
- Save

2. Create a dropbox to receive content.

- Navigate to: Workgroups
- Click on “Create New”->“Dropbox” (not “Workgroup”)
- Set the same name as in step 1: ProviderA_ingest
- optional instructions may be set
- select the metadata profile created in step 1.
- Click “Create”
- Then, external submitters and internal recipients can be invited

When new packages are received from submitters, files will either remain in the dropbox, or be moved to the appropriate workflow hot folder, depending on submitter selection.

9 Troubleshooting

Depending on the configuration file, the tool generates a log file in the same folder or the tool redirects logs to “stdout”.

Faspex provides stdout/stderr output for all post processing scripts used. To consult those logs, in the Web UI, navigate to Server->Post-Processing, then click on the “log” button on the same line as the post processing script. Faspex will provide a list of logs for each execution of the script. Clicking on the date will display all provided metadata as well as stdout/stderr for the script.

To get standard logs inside a single file in normal level, just dont set any value for configuration parameters: `logger` and `loglevel`

To get directly debug level information inside Faspex, the the following values in the configuration file: `logger:stdout` and `loglevel:0`

10 Hack: removing actions from labels in package form

This postprocessing script avoids external configuration by embedding actions in metdata values. Nevertheless, one may want to hide those parameters from the end user. This is not supported natively by Faspex, nevertheless, it is possible to “hack” Faspex in a relatively controlled manner. If using this trick, please first save the modified source file. And, in case of problem with Faspex, revert to the original before isuing a ticket to Aspera Support. The procedure is as follows:

- Identify the file: <faspex root>/app/models/metadata_field.rb
- Typically, <faspex root> is C:Files (x86)or /opt/aspera/faspex
- Make a copy of this file (for instance, add extension “.orig” or “.save”). Important: do not leave a copy of the original file with the “.rb” extension, else methods would be overriden with old definition.
- Edit the text file and look for the following code block near the end of the file:

```
def self.select_fields(options, required = false)
  fields = options.map{|f| [f, f]}
  fields = [['<Select an option>', '']] + fields if !required
  fields
end
```

- Change the second line like this:

```
  def self.select_fields(options, required = false)
    fields = options.map{|f| [f.gsub(/\(.*\)/,''), f]}
    fields = [['<Select an option>', '']] + fields if !required
    fields
  end
```

It consists in adding the code `.gsub(/\(.*\)/,'')` just after the occurence of `[f`.

After the modification, restart Faspex Mongrels by executing this command line:

```
asctl faspex:mongrel:restart
```

To revert back to the original:

- restore the original file
- restart mongrels like above

This change does not survive an upgrade. So, after an upgrade re-iterate the operation (save file, modify, restart mongrel).

11 Access rights to folders

The post-processing scrip[t is executed by the background process: `Faspex Background`. On windows, this process is a service run by a system user. The system user running the service shall have access to files to be moved and destination folder. In particular, if the folder is in a windows Shares in a domain, the user shall be a domain user with access to these folders.

12 Support

The postprocessing script is provided “as is” and is not supported by Aspera or IBM. If a problem occur with Faspex and the “hack” was applied, it is best to revert to the original source code to rule-out that the problem comes from the change.